DOI:10.15923/j.cnki.cn22-1382/t.2018.5.12

基于 Hadoop 平台的改进 KNN 分类算法并行化处理

马 莹, 赵 辉*, 崔 岩

(长春工业大学 计算机科学与工程学院, 吉林 长春 130012)

摘 要: 首先利用 K-medoids 聚类算法对训练样本集进行剪裁,去除相似度程度较低的样本。结合 Hadoop 平台的 MapReduce 框架,采用改进 KNN 分类算法对数量不等的测试样本集在不同节点上进行了加速比并行化计算。实验结果表明,该方法的计算时间和分类效率均有改善。

关键词: K-最近邻; K-medoids 聚类算法; Hadoop 平台; 并行化

中图分类号: TP 301.6 **文献标志码:** A **文章编号:** 1674-1374(2018)05-0484-06

Parallel processing of improved KNN classification algorithm based on Hadoop platform

MA Ying, ZHAO Hui*, CUI Yan

(School of Computer Science and Engineering, Changchun University of Technology, Changchun 130012, China)

Abstract: K-medoids clustering algorithm is used to tailor the training sample set for removing the samples with low degree of similarity. Parallel acceleration rate calculation for different samplings at different nodes are carried out with the improved KNN classification algorithm under MapReduce framework of the Hadoop platform. Experimental results show that the classification efficiency and calculation time are improved.

Key words: K-nearest neighbor; K-medoids clustering algorithm; Hadoop platform; parallelization.

0 引 言

随着科技的快速发展和大数据时代的到来,各领域的社交媒体都在时刻产生大量的数据,而这些数据都存在着潜在的价值[1]。当前,数据挖掘作为发现数据库中有价值数据的关键技术,引起了广大学者的高度关注[2]。数据挖掘是指从庞大的数据量中发现隐藏在其中有价值的数据信息

收稿日期: 2018-08-25

基金项目: 吉林省教育厅"十二五"科学技术研究基金资助项目(2014132)

作者简介: 马 莹(1993—),女,汉族,吉林长春人,长春工业大学硕士研究生,主要从事自然语言处理、智能计算方向研究,E-mail:1300568307@qq.com. *通讯作者:赵 辉(1972—),女,汉族,吉林敦化人,长春工业大学教授,博士,主要从事智能计算、搜索引擎方向研究,E-mail:412600729@qq.com.

的过程，在市场分析、信息统计、科学探索等方面都得到了广泛应用。常用的传统分类算法有：支持向量机（Support Vector Machine，SVM）、K-最近邻（K-Nearest Neighbors，KNN）、朴素贝叶斯（Naive Bayes，NB）等。其中 KNN 分类算法有着思想简单、理论成熟、易于实现、准确度高等优点，因此被广泛应用于各领域的数据挖掘中。但是传统的 KNN 分类算法也存在着以下缺点：

1）传统的 KNN 分类算法作为文本分类中被广泛应用的算法之一，在分类过程中，要计算每一个测试样本与训练样本集中每一个点相似度或距离，因此，在此过程中会由于计算量庞大而耗费大量的时间，最后导致分类速度减慢，算法的时间复杂度增高，分类效率降低。

2）如果训练样本集处于不均匀状态，那么最终会导致分类的结果不准确。然而，随着各领域的数据量不断增加，传统分类算法已经不能满足当前的数据分析需求，因此，提高算法的分类时间和分类准确性是当前数据分类至关重要的问题。

如今，已有很多研究者对 KNN 分类算法进行了相关的探究和分析。任朋启等[3]通过对训练样本集高密度的部分进行了剪裁，并对剪裁后的训练样本集进行了投影寻踪理论，提出了一种改进的 KNN 分类算法——IKNN 分类算法，从而提高了分类的准确性。邓振云等[4]通过引入重构和局部保持投影技术，提出了基于局部相关性的 KNN 分类算法，从而提升了 KNN 分类算法的分类效率。涂敬伟等[5]通过将 KNN 分类算法与 MapReduce 框架相结合，提出了一种在 MapReduce 框架上实现 KNN 分类并行化计算的研究，研究结果表明，此方案有着良好的加速比和可扩充性。

传统的 KNN 分类算法存在着在分类过程中会耗费大量的时间去计算样本间相似度或距离的情况，为了解决此问题，文中首先使用 K-medoids 聚类算法对训练样本集进行了剪裁，去除了样本中相似度较低的部分；然后结合 Hadoop 平台中的 MapReduce 框架实现数据的并行化处理，使其在多个节点上进行运算，从而降低了算法的时间复杂度，提升了算法的运行速度。

1　Hadoop 平台

Hadoop 平台最核心的设计是 HDFS（Hadoop Distributed File System）[6] 和 MapReduce[7]。HDFS 是 Hadoop 的一个子项目，是数据分布式操作的基础，它是以流数据访问模式来存储超大文件，适合运行在通用硬件上[8]，具有高可靠性、高可扩展性等特征。而 MapReduce 是 Hadoop平台中一个并行计算的框架模型，为大量的数据提供并行化模式。它具有数据分类、计算工作调度、系统优化等主要功能[9]。MapReduce 编程模型的计算过程分为两部分：Map 函数和 Reduce函数，用户只要实现 Map 函数和 Reduce 函数，就可以完成分布式计算[10]。具体分为以下几部分：

1）用户提交数据信息后，MapReduce 首先读取 HDFS 中的数据信息，并将其分割成 M 个 split，然后将划分的信息传送给 JobTacker，并通过 Fork 创建 master 和 worker。

2）在 Map 阶段，Map 任务数量是由片段 M 决定的，与 split 是一一对应的，并且每个 Map 任务之间是相互独立的。

3）Map 函数和 Reduce 函数都是以＜key，value＞键值对的方式进行输入和输出，Map 函数从得到的数据信息中提取出＜key，value＞键值对作为输入，然后进行操作得出以＜key，value＞键值对方式的中间结果，Map 函数产生的中间结果被缓存在内存中。

4）在 Reduce 函数阶段，将获取的中间结果按照 key 值进行遍历排序，使得同一 key 值所对应的 value 值在一起。

5）将中间结果以＜key，value＞键值对的方式传递给 Reduce 函数操作，并且将中间结果的 value 值进行归并，最终以＜key，value＞键值对的方式进行输出。MapReduce 编程模型如图 1 所示。

2　KNN 算法描述

KNN 分类算法是在 1968 年由 Cover 和 Hart 提出的，是数据挖掘中最简单的方法之一，在语言处理、图像分析、信息检索等方面都得到了广泛的研究和应用[11]。它的基本原理是：

首先对待测试样本集与训练样本集中的每一个点进行相似度或距离计算，然后找出与测试样本距离最近或是相似度最高的 K 个训练样本，并以此作为待测试样本的 K 个近邻，最后根据 K 个近邻所属的类别对待测试样本进行划分归类，找到最终属于哪一类别。

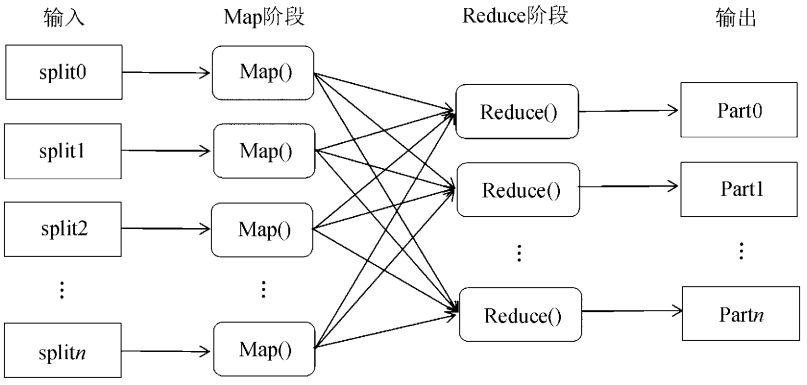

图1 MapReduce编程模型

KNN分类算法具体实现步骤：令 D 代表训练样本集，其中 N 代表 D 的样本类别个数，表示形式为 $\{C_1, C_2, \cdots, C_N\}$，$M$ 代表训练样本集数量，n 代表特征向量的维度阈值，$\boldsymbol{d}_i$ 代表 D 中的一个样本的特征向量，表示形式为 $\{x_{i1}, x_{i2}, \cdots, x_{ij}, \cdots, x_{in}\}(0<i\leqslant M)$，其中 x_{ij} 表示 $\boldsymbol{d}_i$ 的第 j 维的权重 $(0<j\leqslant n)$，$\boldsymbol{d}=\{X_1, X_2, \cdots, X_j, \cdots, X_n\}$ 代表待测试样本的特征向量方式，X_j 表示 $\boldsymbol{d}$ 的第 j 维的权重 $(0<j\leqslant n)$，相似度的计算公式如下：

$$\mathrm{Sim}(d, d_i)=\frac{\sum_{j=1}^{n}(X_j x_{ij})}{\sqrt{\sum_{j=1}^{n}(X_j^2)}\sqrt{\sum_{j=1}^{n}(x_{ij}^2)}} \quad (1)$$

计算出待分类样本与训练样本集中每一个点的相似度后，通过式(1)找到了待分类样本的相似度最高的 K 个最近邻样本，再通过下式计算出待分类样本归属于每一个类别的权重，最后将待分类样本按照权重大小进行划分，分配到权重最大的类别中。

$$W(d, C_j)=\sum_{i=1}^{K}\mathrm{Sim}(d, \boldsymbol{d}_i)\,y(\boldsymbol{d}_i, C_j) \quad (2)$$

其中，$y(d_j, C_j)$ 为类别属性函数，如下式：

$$y(\boldsymbol{d}_i, C_j)=\begin{cases}1, & \boldsymbol{d}_i \in C_j\\ 0, & \boldsymbol{d}_i \notin C_j\end{cases} \quad (3)$$

3 KNN分类算法的改进

KNN分类算法虽然易于理解，计算简单，但是也存在着一定的缺点，文献[12]指出在分类过程中需要计算待测试样本与训练样本集所有点的距离，这样浪费了大量的分类时间，从而减少了KNN分类算法的分类效率。针对传统KNN分类算法在分类过程中存在计算量较大的问题，文中将K-medoids聚类算法引入到训练样本集中进行聚类剪裁，从而降低相似度的冗余计算。

K-中心聚类算法，即K-medoids聚类算法，是一种被广泛应用于数值统计、统计学、数据探索、医学诊断等重要领域的传统聚类方法，它具有数据简洁、计算简单、高健壮性等特性。

令训练样本集为 D，其中 D 的样本类别数为 $N\{C_1, C_2, \cdots, C_N\}$，训练集中共包含的样本数为 M，训练样本集剪裁方式如下：

1)K-medoids初始簇心的选择和优化。

①对训练样本集 D 进行划分，将其分为 m 个簇，其中 $m=3*N$。

②为每一个簇随机选择一个点作为簇心 O_i $(0<i\leqslant m)$。

③计算出训练样本集 D 中所有点到簇心 O_i 的相似度，并按照相似程度对其进行分配。

④在每一个簇内，首先令簇内的每一个点作为簇心，然后计算簇心到其他簇的相似距离和，最终选择相似距离和最小的簇心作为簇内新的簇心 O_i。

2)对替换簇心搜索进行优化。

①选择一个未被选取的簇心 O_i 替换簇心集 A，从而不再使用全局的非簇心集，而是使用距离簇心 O_i 最近的 j（j 代表迭代次数，$0<j\leqslant m$）个簇含有的非簇心文本区。

②计算出在簇心集 A 中选取的没有被选择过的非簇心 Q_i 与簇心 O_i 之间的平方误差之差，并将计算结果写在集合 Q 中，直至簇心集 A 中的所有非簇心都对比过。

③如果集合 Q 最小值小于0，用集合 Q 中最小值所对应的非簇心替换原簇心，替换后得到新

的簇心集合。然后把剩下的文本分给对应的簇内，其中此簇的簇心相似度最大。最后从步骤①重新开始进行计算。

④如果集合 Q 的最小值大于或等于 0，那么停止计算，最终取得 m 个聚类簇心。

3)训练样本集剪裁。

统计待分类样本与 m 个聚类簇心的相似度，如果 $\mathrm{Sim}(D,O_i)$小于 T_i(其中，T_i 代表第 i 个簇的簇内阈值，即簇内样本与簇心的最小相似度)，表示待分类样本与簇内样本的相似程度比较低，因此可以把该簇内包含的样本进行剪裁，否则把该簇内包含的样本添加到新的训练样本集中。

4　改进 KNN 算法的并行化处理

4.1　基于 Hadoop 平台的改进 KNN 分类算法并行化处理

KNN 分类算法主要是获取 k 个最近邻，要实现这一过程必须通过大量的距离计算。设计并行计算可以明显地减少分类时间，提高分类效率。基于 Hadoop 改进的 KNN 分类算法实现 MapReduce 框架并行计算的基本思路是：首先将训练样本集分配给不同的节点进行 Map 函数操作计算形成键值对的形式，完成数据记录到训练样本距离的相似度计算以及相关的排序操作，相似度计算采用的是标准的欧式距离计算度量，结果存入到 Context 集合中，在此过程中每一个 Map 函数之间操作过程是没有关联的；然后将 Map 函数操作的计算结果作为中间结果是以＜key，value＞键值对的方式输入给 Reduce 函数；最后 Reduce 函数接受传入的各个 Map 节点的操作结果，其中输入数据的方式是＜key，value＞键值对的形式，根据用户指定的最近邻树 k 进行排序，最终归并输出。

文中的 Map 函数和 Reduce 函数的相关伪代码如下：

Map 函数的伪代码：

```
Input:Text key,Vector value
Output: <Text,Vector>Context context
Begin:
  For i=0 to n (training dataset) do
    t=FindCatalog(i);
      For all k∈testfile do
        Distance=EuclideanDistance(k,ji);
        Context.write(key,vector(t,Distance));
      End For
  End For
End
```

Reduce 函数的伪代码：

```
Input:Text key,Vector value
Output:Text key,Vector value,Context context
Begin:
  For all key and value do
    Array List(vcetor(t,value));
    Sort(Array List);
    New ArrayList result;
    If k<ArrayList.size then
      For i=0 to k do
        result.add(key,ArrayList.get(i));
    Else
  System.out.println("no sufficient training samples");
      Context.write(key.Tradition KNN(result))
      End for
    End for
End
```

4.2　时间复杂度分析

传统的 KNN 分类算法的时间复杂度为 $O(r_1r_2)$，r_1 为训练集样本数量，r_2 为测试集样本数量。文中对训练集数据进行了聚类剪裁，降低了相似度的冗余计算，从而生成新的训练集，数目为 r_3，此时时间复杂度为 $O(r_2r_3)$，在改进的 KNN 分类算法之上并结合 MapReduce 框架对 KNN 算法实现了并行化处理，时间复杂度为 $O(r_2r_3/n)$，n 为节点个数。由于 $r_1>r_3$，所以基于 Hadoop 平台改进的 KNN 分类算法的时间复杂度与传统的 KNN 分类算法的时间复杂度关系为 $O(r_2r_3/n)<O(r_1r_2)$。

5　实验结果与分析

5.1　实验环境与数据集

实验平台为 12 台虚拟机构成的集群，CPU 型号为 Intel(R) Core(TM) i7-4790 CPU @ 3.60 GHz，内存为 8 G，Hadoop 版本为 2.6.0。

文中采用微博文本作为实验数据进行分类，语料中分为正向情感和负向情感两个类别，每个类别中有 5 000 篇文本，共计 10 000 篇文本。根据需求从每个文本类别中随机抽取 1 000 篇文本，共计 2 000 篇文本作为训练集。为了实验的有效性和全面性，将剩余的数据分为不同规模的测试集，见表 1。

表1 测试集

测试集	规模大小	详情
A1	1 000	从各类中随机抽取500篇
A2	5 000	从各类中随机抽取2 500篇
A3	8 000	完整测试集

5.2 实验及结果

针对算法性能方面的评价指标，比较加速比、正确率 P 与运行时间 t，其中加速比公式为：

$$\text{Speedup}(g)=\frac{\text{一个节点上使用的时间}}{m\text{ 个节点上使用的时间}} \tag{4}$$

正确率公式为：

$$P=\frac{\text{正确分类的文本数}}{\text{总文本数}} \tag{5}$$

实验1：在单一节点上，以正确率 P 与运行时间 t 作为评价指标，对传统的KNN分类算法、基于K-medoids改进的KNN分类算法二者之间进行了比较。比较结果如表2和图2所示。

表2 两种算法的比较

测试样本集	传统的KNN分类算法		基于K-medoids改进的KNN分类算法	
	P/%	t/s	P/%	t/s
A1	81.1	257	80.5	205
A2	82.7	392	82.5	341
A3	88.2	595	90.6	460
平均	84.0	415	84.5	335

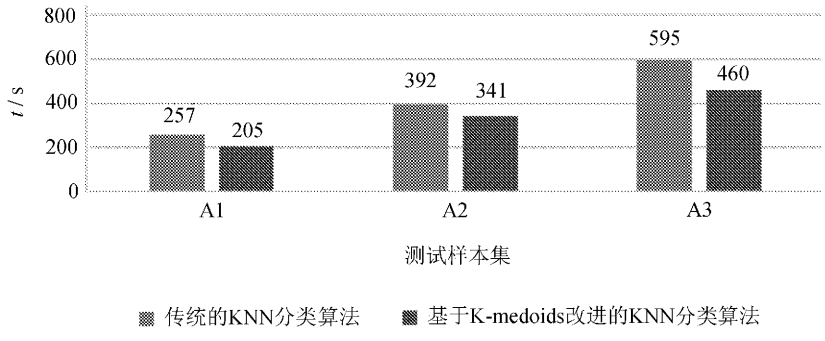

图2 两种算法的运行时间的比较

从表1和表2可以看出，测试样本集A1中的信息数据量比较少，文中基于K-medoids改进的KNN分类算法的正确率比传统的KNN分类算法降低了0.6个百分点，这是由于通过K-medoids聚类算法剪裁后的算法为近似算法，当数据信息量较少时，会对分类结果的正确率产生一定的影响；但是在时间方面上，由图2可以明显看出，基于K-medoids改进的KNN分类算法比传统的KNN分类算法缩短了13%～22%，而且随着数据量的增大，计算时间也缩短的越为明显，这是由于剪裁之后，降低了相似度的冗余计算。

实验2：加速比主要是用来权衡一个系统的可扩充性。在基于Hadoop平台改进的KNN分类算法下，针对节点个数不同时，对不同规模的测试样本集之间的加速比进行了比较，结果见表3。

表3 基于Hadoop平台的改进KNN分类算法

测试集	节点数/个				
	1	20	30	40	50
A1	1.0	1.97	3.01	4.05	5.08
A2	1.0	2.01	3.05	4.09	5.12
A3	1.0	2.09	3.13	4.18	5.20

根据表3可以看出，基于Hadoop平台改进的KNN分类算法的加速比随着节点个数的增加而保持着线性上升，并在节点个数增加到30个时，加速比明显升高。多个节点可以明显缩减分

类过程中所需要的时间，这表明在操作 KNN 分类算法时 Hadoop 平台上具有较好的加速比。由此可见，当节点个数逐渐增加时，加速比增长的速度也会更快。

5.3　实验结果分析

文中主要研究基于 Hadoop 平台的改进 KNN 分类算法的并行化处理，首先通过 K-medoids聚类算法对传统的 KNN 分类算法进行剪裁，然后通过 Hadoop 平台中的 MapReduce 框架实现数据并行化计算。从表 1 和表 2 中可以看出，在选择完整的测试集时，基于 K-medoids 改进的 KNN 分类算法比传统的 KNN 分类算法在运行时间上缩短了 150 s；从表 3 可以看出，加速比随着节点个数的增加而快速增长，特别是节点个数在 30～50 之间的时候，产生以上原因是因为 Hadoop 平台的关键技术之一——MapReduce 框架，在此框架上实现数据的并行化计算，提高了在分类过程中的运行时间。

6　结　语

在大数据时代通过对传统 KNN 分类算法的分析和研究，文中提出了基于Hadoop平台改进的 KNN 分类算法的并行化处理。首先将 K-medoids聚类算法引入到传统的 KNN 分类算法中，对 KNN 分类进行剪裁，去除了相似程度较低的文本，然后运用 Hadoop 平台中的 MapReduce框架对文本数据实现并行化计算，在分类过程中减少了算法的时间复杂度，提高了分类速度。实验结果表明，选择基于 Hadoop 平台的改进 KNN 分类算法对文本分类在时间复杂度方面取得了良好的分类效果，提高了分类效率，并且能够适用于当前的大数据环境。

参考文献：

[1] 姜奇平.大数据时代到来[J].互联网周刊，2012(2)：6-10.

[2] 柴艳妹，雷陈芳.基于数据挖掘技术的在线学习行为研究综述[J].计算机应用研究，2018，35(5)：1287-1293.

[3] 任朋启，王芳，黄树成.一种改进的文本分类算法[J].电子设计工程，2017，25(18)：1-5.

[4] 邓振云，龚永红，孙可，等.基于局部相关性的 KNN 分类算法[J].广西师范大学学报：自然科学版，2016，34(1)：52-58.

[5] 涂敬伟，皮建勇.基于 MapReduce 和分布式缓存的 KNN 分类算法研究[J].微型机与应用，2015，34(2)：18-21.

[6] Doug Cutting. Apache Hadoop YARN[EB/OL]. [2018-08-25]. Http://hadoop.apache.org/.

[7] Dean J，Ghemawat S. MapReduce：simplified data processing on large clusters [J]. Communications of the ACM，2008，51(1)：107-113.

[8] 郭博洋.大数据 hadoop 的来源与介绍[J].计算机产品与流通，2017(10)：155.

[9] 王泽儒，王红梅，李芬田.基于 Hadoop 的 2FP-Growth 算法[J].长春工业大学学报，2018，39(2)：150-155.

[10] 夏靖波，韦泽鲲，付凯，等.云计算中 Hadoop 技术研究与应用综述[J].计算机科学，2016，43(11)：6-11，48.

[11] 毋雪雁，王水花，张煜东.K 最近邻算法理论与应用综述[J].计算机工程与应用，2017，53(21)：1-7.

[12] 樊存佳，汪友生，边航.一种改进的 KNN 文本分类算法[J].国外电子测量技术，2015，34(12)：39-43.